Evolved finite state controller for hybrid system in reduced search space

Dupuis, Jean-Francois; Fan, Zhun

Published in:
IEEE/ASME International Conference on Advanced Intelligent Mechatronics, 2009. AIM 2009.

Link to article, DOI:
10.1109/AIM.2009.5229908

Publication date:
2009

Document Version
Publisher's PDF, also known as Version of record

2009 IEEE/ASME International Conference on Advanced Intelligent Mechatronics
Suntec Convention and Exhibition Center
Singapore, July 14-17, 2009

Evolved Finite State Controller For Hybrid System In Reduced Search Space

Jean-François Dupuis and Zhun Fan

Abstract— This paper presents an evolutionary methodology to automatically generate finite state automata (FSA) controllers to control hybrid systems. The proposed approach reduce the search space using an invariant analysis of the system. FSA controllers for a case study of two-tank system have been successfully obtained using the proposed evolutionary approach. Experimental results show that these controllers have good performance on the set of training targets as well as on a randomly generated set of validation targets.

I. Introduction

Mechatronic systems are the complete integration of mechanics, electronics and information processing. Tight integration of these domains make them highly dependent on each other. Design choices in one domain affect the performance of the other domains. Therefore, design of such a system usually requires iterations in each domain [1] in order to find an optimal balance between the basic mechanical structure, sensor and actuator implementations, automatic digital information processing and overall control. In an effort to automate the generation of mechatronic systems spanning multiple domains, the use of the bond graph [2] representation and genetic programming for search was proposed [3]–[8]. In this previous work, interesting results on a variety of case studies were presented. However, in all of these examples of automated design, the information processing capability was quite limited. In fact, all the case studies presented were of time-driven systems. The addition of an event-driven controller to a mechatronic system results in a more intelligent device. Mixing both time-driven and event-driven control in a hybrid controller increases the design complexity achievable in comparison to current automated design systems.

Such hybrid systems may be viewed as an extension of a classical time-driven system, typically modelled through differential or difference equations, with occasional discrete events causing a change in its dynamic behaviour. When such an event takes place, the system is thought of as switching from one operating mode to another. Hybrid or switched bond graph representations [9], [10] have been proposed to model physical dynamic systems with discontinuity. This hybrid system representation extends the normal bond graph by adding a switch element that can act as a flow source or as an effort source according to the current system state or controller action. A supervisory control system [11], [12] can thus extend the decision making capability of a normal bond graph in a compact way. The resulting hybrid system can be represented as in Figure 1 [12], where the hybrid bond graph acts as the plant and the controller manages the state of the switch element in the bond graph. The interface acts as a translator between the continuous space and the discrete controller space, whereas the switches in the hybrid bond graph act as the actuators. The generator is usually part of the controller design as it needs to generate meaningful symbols according to the state of the plant.

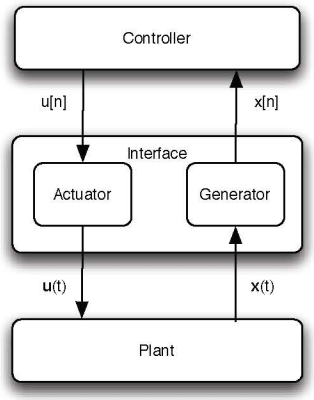


Fig. 1. Hybrid control system.

Several techniques have been proposed through the years to automate the design of controllers. Neural networks with fixed and open topologies [13] are often seen for time-driven systems. However, for supervisory controllers, finite state automata (FSA) are usually better at representing the logical relationships in the system. The interest in evolution of FSA is a very old one, having started almost 50 years ago with the work of Larry Fogel [14], but is still active today [15]–[17].

This article uses an invariant analysis of the system in order to reduce the search space of an evolutionary algorithm that generate a FSA controllers for hybrid systems. A two-tank system is used as a study case to demonstrate of the feasibility of the approach. The remainder of the paper is organized as follows the next section describes the two tanks systems and its controller. Then Section III presents the evolutionary setup that was used to conduct the experiments for which results are analysed in Section IV. Finally, Section V concludes the paper with perspectives and future work

II. Two-Tank System

Multiple-tank systems are often encountered in the research literature concerning non-linear multi-variable feedback control, as well as in fault diagnosis literature [18], [19].

Both authors are with the Technical University of Denmark, Nils Koppels Alle, Building 426, 2800 Kgs. Lyngby, Denmark jfd@mek.dtu.dk, zf@mek.dtu.dk

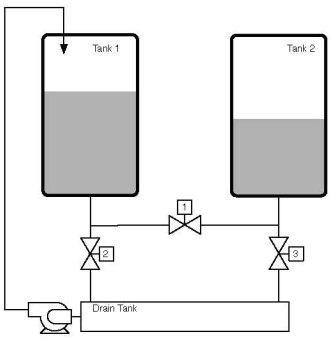


Fig. 2. The two-tank system.

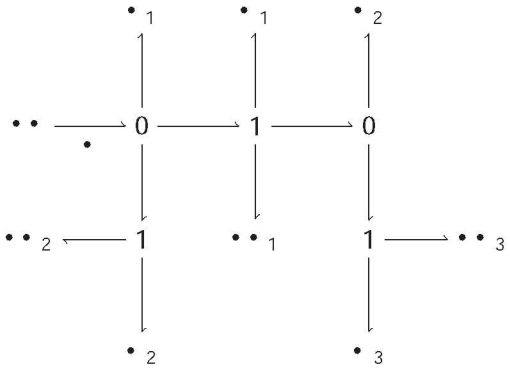


Fig. 3. The hybrid bond graph of the two-tank system.

The mechanical simplicity and the ease of getting physical insight into the system behaviour, combined with the achievable control complexity, make the multi-tank problem a very attractive testbed. Therefore, a two-tank system was defined to test the controller synthesis methodology presented in this article. Figure 2 shows the actual configuration. A pump is continuously filling the first tank at a constant flow rate, and a set of valves allows drainning each tank independently and also allows bidirectional transfer from one tank to the other.

The hybrid bond graph of this two-tank system is shown in Figure 3. The pipe and valve restrictions are represented by the resistive components, • , while the tanks are represented by the capacitances, • . The valves are simply modelled by switch components, • • . These switches act as a 0-flow source or a 0-effort source, depending on their state. A 0-flow source imposes a flow equal to zero at the junction connected to it, therefore the valve is said to be closed as no fluid is able to pass through it. A 0-effort source indicates that the switch does not impose any restriction on the flow. The valve is then said to be open.

The vector state equation of the two-tank system can be obtained from the hybrid bond graph :

$$\begin{bmatrix} \dot{p}_1 \\ \dot{p}_2 \end{bmatrix} = \begin{bmatrix} \frac{1}{C_1}\left(-\frac{1}{R_1}-\frac{1}{R_2}\right) & \frac{1}{C_2}\left(\frac{1}{R_2}\right) \\ \frac{1}{C_1}\left(\frac{1}{R_2}\right) & \frac{1}{C_2}\left(-\frac{1}{R_2}-\frac{1}{R_3}\right) \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u \tag{1}$$

in which • denotes the state vector and • the input vector.

The level • of tank • can be obtained from the state variables :

$$h = \frac{p}{\rho g} \tag{2}$$

where • is the fluid density and • is gravity. This system of equations can also be expressed using the matrix formulation :

$$\dot{x} = \begin{bmatrix} \left(-\frac{1}{R_1 C_1}-\frac{1}{R_2 C_1}\right) & \frac{1}{R_2 C_2} \\ \frac{1}{R_2 C_1} & \left(-\frac{1}{R_2 C_2}-\frac{1}{R_3 C_2}\right) \end{bmatrix} x + \begin{bmatrix} 1 \\ 0 \end{bmatrix} u$$
$$y = \begin{bmatrix} \frac{1}{\rho g A_1} & 0 \\ 0 & \frac{1}{\rho g A_2} \end{bmatrix} x \tag{3}$$

Therefore, the equation in state space form for this two-tank system is :

$$\begin{aligned} \dot{x} &= A x + B u \\ y &= C x \end{aligned} \tag{4}$$

A. Invariant analysis

In the proposed approach, the natural invariants of the dynamical system describe by equation (1) are analysed. This analysis aim at identifying the control policies that can partition the continuous state space. The policies that don't provide partitionning hypersurfaces of the state space will be discarded from the valid control policy set.

A set of candidate partitionning hypersurfaces for a given control policy, u_i, can be identified by solving the characteristic equation of the system :

$$\frac{\dot{p}_1}{\dot{p}_1 = f_1(x, u_i)} = \frac{\dot{p}_2}{\dot{p}_2 = f_2(x, u_i)} \tag{5}$$

If these hypersurfaces can be found, this means that a stream leading to the target zone can be defined using the actuator states defined by that control policy. However, if no hypersurface can be defined for a given control policy, that would means that the behavior of the system cannot be clearly establish when applying the associated actuator states.

Solving the characteristic equation of the two-tank system was only possible for five of the eight possible actuator states. Therefore, three of the eight control policy were rejected:

$$u = [0\,0\,0\,0\,0\,0] \qquad u = [0\,0\,0\,0\,0\,0] \qquad u = [0\,0\,0\,0\,0\,0] \tag{6}$$

This reduction of the number of control policy reduce significantly the search space needed to be explored by the evolutionary algorithm.

B. FSA controller

The controller trained for this two-tank system is a FSA having a fixed number of five states as established in the previous section. Each state corresponds to a possible switch configuration in the hybrid bond graph, where • . is associated to the bond graph switch • • ..

The input symbols are generated by the interface when the state variables cross predefined surfaces in the state space. These surfaces are separated into two sets, one for each tank, and are defined as follow :

$$\begin{aligned} \Sigma_1 &= \{\sigma_{1,1}, \sigma_{1,0}, \sigma_{1,-1}\} \\ \Sigma_2 &= \{\sigma_{2,1}, \sigma_{2,0}, \sigma_{2,-1}\} \end{aligned} \tag{7}$$

in which •. is the desired level for tank • and • is the tolerance about the target. Each set separates the space into three regions depending on whether the level of the tank • is above, below or at its target. Then nine symbols are formed from the logical conjunction of the two sets.

The transitions used in this implementation of the FSA do not have any actions associated with them; they simply specify what the new state will be, in reaction to the input symbol received. Therefore, the controller can be expressed as a simple matrix with the states as row indices and the input symbols as column indices. The elements of the matrix then specify the next state to which the FSA should move. An initial state need also to be defined outside this matrix to complete the definition.

III. Experimental setup

For the experiments reported, the two-tank controller was trained to keep the fluid levels of the tanks within their target regions.

A. Fitness evaluation

When evaluating the fitness of the evolved controllers, for each simulation case, the system of equations 1 is integrated for a period of 15 seconds. An objective function •••• is computed for each tank at the end of the simulation, based on the level errors :

$$\bullet \quad \bullet\bullet\bullet\bullet \quad \bullet\bullet\bullet\bullet \quad \bullet\bullet\bullet\bullet \quad \bullet\bullet\bullet\bullet^{2} \quad \bullet\bullet \tag{8}$$

Later, the fitness of this simulation case is defined as the fitness of the tank with the worst error:

$$\bullet \quad \bullet \quad \frac{\bullet}{\bullet \quad \bullet\bullet\bullet\bullet \quad \bullet\bullet\bullet\bullet} \tag{9}$$

Looking at the tank with the worst error proved to be a more successful approach than looking at the average fitness of the two tanks. When looking at the average, the incentives to reach the target were not strong enough, and the evolution was ceasing after finding compromises between the errors of the two tanks. Most of the time it was observed that one of the tanks sacrificed itself in order to get excellent performance by the other. The really poor performance of the sacrificed tank was then compensated for by the performance of the other tank, yielding a high average. However, looking at the worst tank disallows such behavior and enforce having good performance of both tanks.

B. Multiple simulation cases

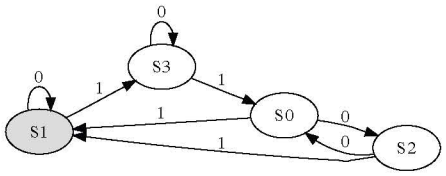
In order to obtain controllers that generalize to cases outside the training set, the controller is tested on several simulation cases. For each case the fitness is computed as defined in the previous section and again, the worst case is used as the fitness processed by the selection operator.

	0	1
0	2	1
1	1	3
2	0	1
3	3	0

(a) Transition table.

10 01 01 11 00 01 11 00 01

(b) Genotype



(c) Phenotype

Fig. 4. Encoding example

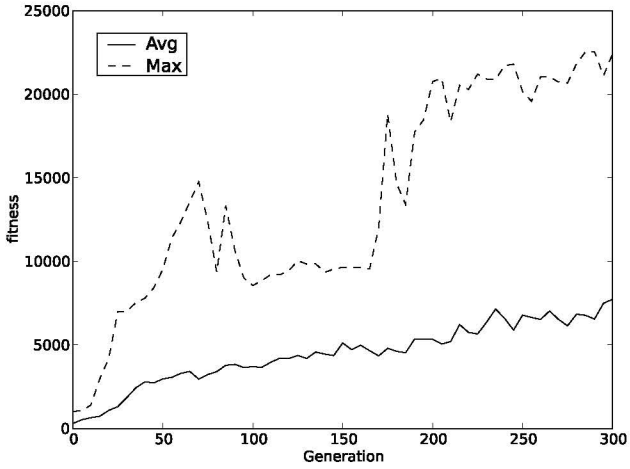


Fig. 5. Population best and average fitness using eight states.

C. Genetic algorithm implementation

The evolution of the controller uses a simple genetic algorithm (GA) with standard one-point crossover, bit-flipping mutation and tournament selection. The transition table matrix described in section II-B is encoded in a bit string with three bits per element, with three extra bits at the end to define the initial state. The evolved bit string is thus 219 bits long. The Figure 4 shows the proposed encoding approach used in a simpler case with only four states and two input symbols. In this example, only two bits per element is used. One can see that the elements of the transition matrix are written row by row to the associated genotype.

The implementation of this GA experiment was done using the Open BEAGLE C++ framework [20] and the evaluation of the fitness was distributed on a cluster of computers using MPI [21].

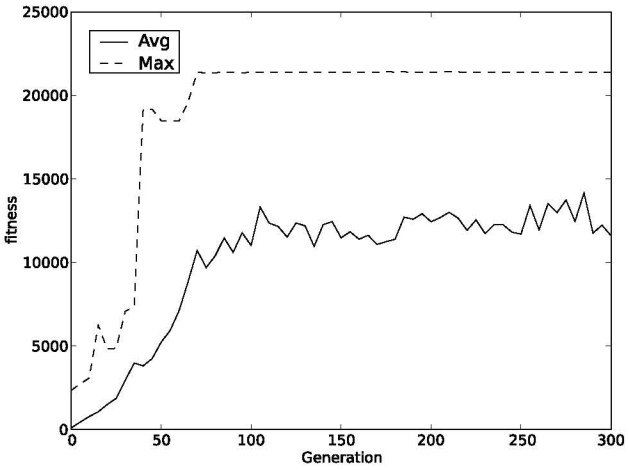


Fig. 6. Population best and average fitness using five states.

IV. Results

The evolutionary algorithm described above was successful in finding controllers that performed well for both the reduced and unreduced controller. The graphs of the fitness as a function of the generation, presented in Figures 5 and 6, show that the fitness of the 5-states controller grows much quicker than the 8-states controller. The reduced search space significantly help at increasing the speed of convergence of the algorithm.

In the end, the performance of both controllers are quite satisfactory. There were still some imperfections that could be corrected, such as the overshoot on the way down of the first tank as seen in Figures 7(b) and 7(e) and the ripples observed in Figure 8(f). On the other hand, both evolved controllers are able to overcome some difficult tasks, such as the one shown in Figure 7(a). In this case, the desired target asks for a level in tank 2 higher that in tank 1. From the inspection of Figure 2, the only way to raise the level in the second tank is to first raise the level in tank 1 and then transfer the fluid between them. Consequently, the level in tank 1 needs to go away from its target in order to help to reach the objective of the other tank. Even though this "enabling" behavior is not rewarded by the fitness function, as tank 1 is accumulating much error during this process, the evolution is able to find this behavior as the best compromise for achieving the best fitness. This kind of task is a difficult type to solve, because when looking at the state space, one must first leave the target region in order to find a path to it.

At the end of the evolution, the best-performing controller, for which the phenotype is shown in Figure 9, was then tested on a set of 20 targets generated at random in order to verify if the evolved controller was able to generalize to other control targets outside its training set. Figure 10 shows the behavior of the controller on the 20 targets of the verification set. As can be seen, the evolved controller generalizes very well on the submitted random targets.

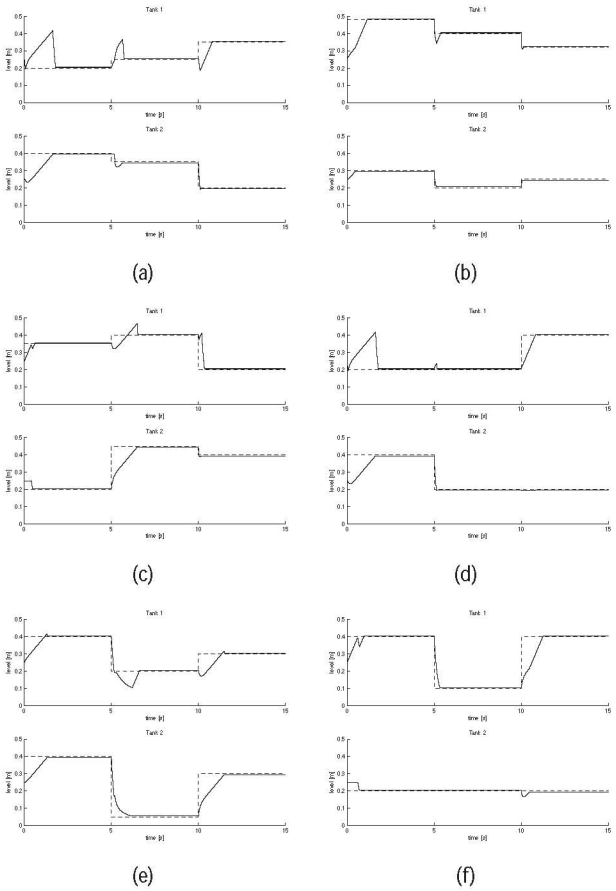


Fig. 7. 8-States controller behavior on the training target of the best individual at the end of the evolution.

V. Conclusion

In this paper, an evolutionary approach to the generation of a FSA controller for hybrid systems was described. A two-tank system was used as a study case. The genetic algorithm implemented successfully evolved controllers showing good performance on a set of training targets as well as on a set of validation targets.

Evolutionary algorithms are again shown to be very powerful search methods. However, the computational power required by the repeated simulation of the controllers make it difficult to scale up. The speed of convergence of the presented algorithm was significantly improved by reducing the search space by analysing the control invariants of the system. More gains might be made by automating the invariant analysis in order to generate a FSA controller directly from the hypersurfaces obtained [12]. Then with significant speed gain, the search space could be extended to include the hybrid bond graph as part of the evolutionary process.

References

[1] Q. Li, W. J. Zhang, and L. Chen, "Design for control-a concurrent engineering approach for mechatronic systems design," Mechatronics, IEEE/ASME Transactions on, vol. 6, pp. 161–169, 2001.

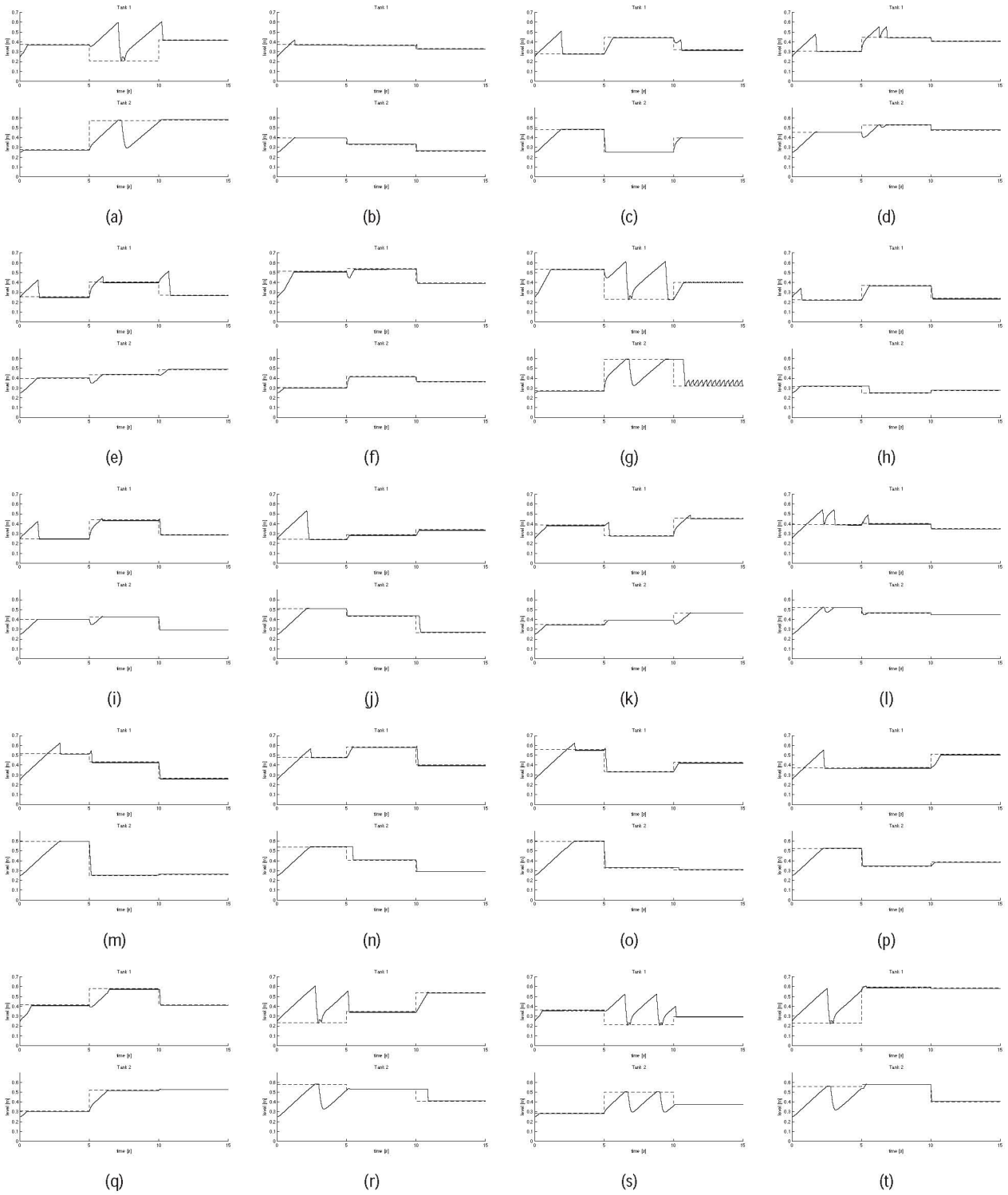


Fig. 10. 5-States controller behavior on the test target after the evolution.

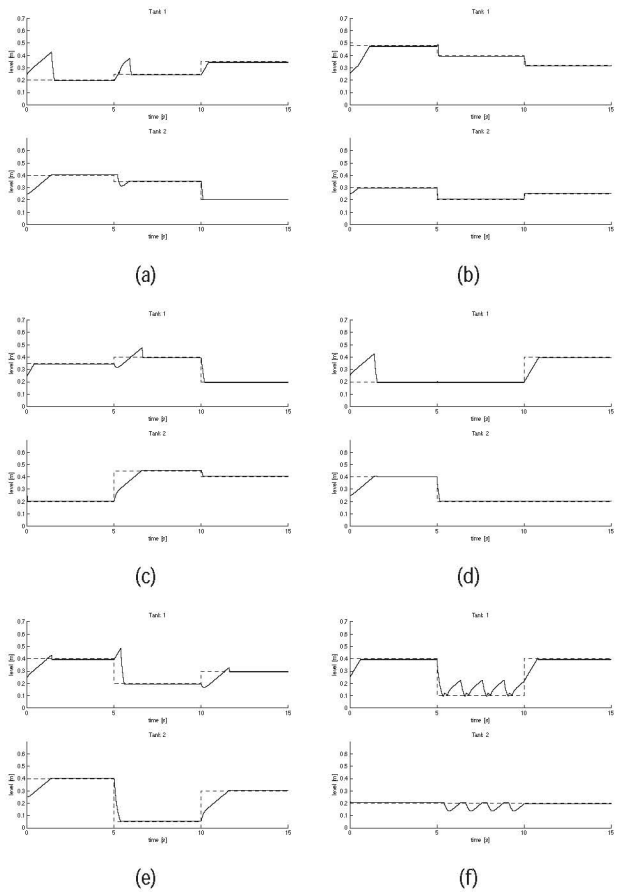


Fig. 8. 5-States controller behavior on the training target of the best individual at the end of the evolution.

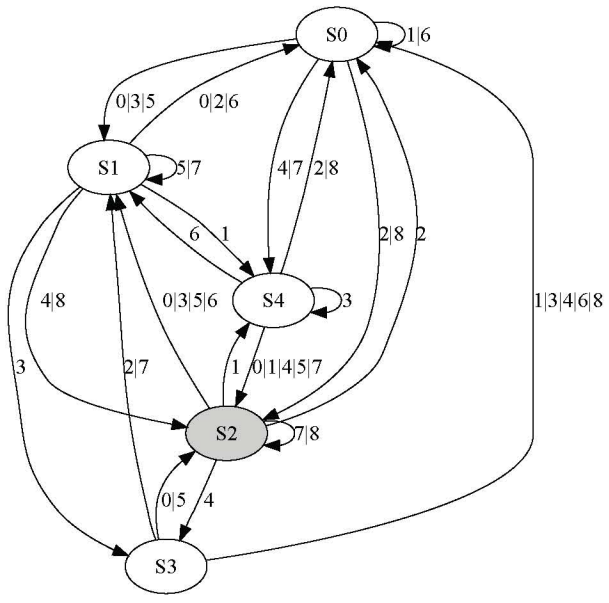


Fig. 9. Phenotype of the best evolved controller. The initial state is indicated by the gray filling and the disjunctive set of transition trigging symbols is marked on the associated transition.

[2] D. Karnopp, D. Margolis, and R. Rosenberg, System Dynamics: Modeling and Simulation of Mechatronic Systems, 4th ed. Hoboken, New Jersey: John Wiley and Sons, 2005.

[3] K. Seo, Z. Fan, J. Hu, E. D. Goodman, and R. C. Rosenberg, "Toward an automated design method for multi-domain dynamic systems using bond graphs and genetic programming," Mechatronics, vol. 13, no. 8-9, pp. 851–885, 2003.

[4] Z. Fan, J. Wang, and E. Goodman, "Exploring open-ended design space of mechatronic systems," International Journal of Advanced Robotic Systems, vol. 1, pp. 295–302, 2004.

[5] J. Wang, Z. Fan, J. P. Terpenny, and E. D. Goodman, "Knowledge interaction with genetic programming in mechatronic systems design using bond graphs," Systems, Man and Cybernetics, Part C, IEEE Transactions on, vol. 35, pp. 172–182, 2005.

[6] J. Wang, Z. Fan, J. P. Terpenny, and E. D. Goodman, "Cooperative body-brain coevolutionary synthesis of mechatronic systems." AI EDAM, vol. 22, no. 3, pp. 219–234, 2008.

[7] Z. Fan, K. Seo, J. Hu, E. D. Goodman, and R. C. Rosenberg, "A novel evolutionary engineering design approach for mixed-domain systems," Engineering Optimization, vol. 36, no. 2, pp. 127 – 147, 2004.

[8] Z. Fan, J. Wang, S. Achiche, E. Goodman, and R. Rosenberg, "Structured synthesis of mems using evolutionary approaches," Appl. Soft Comput., vol. 8, no. 1, pp. 579–589, 2008.

[9] P. J. Mosterman, "Hybrid dynamic systems: A hybrid bond graph modeling paradigm and its application in diagnosis," Ph.D. dissertation, Vanderbilt University, 1997.

[10] J.-E. Strömberg, S. Nadjm-Tehrani, and J. Top, "Switched bond graphs as front-end to formal verification of hybrid systems," Hybrid Systems III, pp. 282–293, 1996.

[11] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," SIAM Journal on Control and Optimization, vol. 25, no. 1, pp. 206–230, 1987. [Online]. Available: http://link.aip.org/link/?SJC/25/206/1

[12] X. Koutsoukos, P. Antsaklis, J. Stiver, and M. Lemmon, "Supervisory control of hybrid systems," in Proceedings of the IEEE, vol. 88, 2000, pp. 1026–1049.

[13] K. O. Stanley and R. Miikkulainen, "Evolving neural networks through augmenting topologies," Evolutionary Computation, vol. 10, no. 2, pp. 99–127, 2002. [Online]. Available: http://www.mitpressjournals.org/doi/abs/10.1162/106365602320169811

[14] L. J. Fogel, "Autonomous automata," Industrial Research, vol. 4, pp. 14–19, 1962.

[15] D. Ashlock, Evolutionary Computation for Modeling and Optimization. Springer New York, 2006, ch. Evolving Finite State Automata, pp. 143–166.

[16] K. Benson, "Evolving finite state machines with embedded genetic programming for automatic target detection," Evolutionary Computation, 2000. Proceedings of the 2000 Congress on, vol. 2, pp. 1543–1549 vol.2, 2000.

[17] B. D. Dunay, F. E. Petry, and B. P. Buckles, "Regular language induction with genetic programming," in Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on, 1994, pp. 396–400 vol.1.

[18] J. Wu, G. Biswas, S. Abdelwahed, and E. Manders, "A hybrid control system design and implementation for a three tank testbed," IEEE Conference on Control Applications, pp. 645–650, 2005.

[19] M. Sainz, J. Armengol, and J. Vehi, "Fault detection and isolation of the three-tank system using the modal interval analysis," Journal of Process Control, vol. 12, no. 2, pp. 325–338, 2002.

[20] C. Gagné and M. Parizeau, "Genericity in evolutionary computation software tools: principles and case study," International Journal on Artificial Intelligence Tools, in-press 2006.

[21] MPI: A Message-Passing Interface Standard, Message Passing Interface Forum, http://www.mpi-forum.org, June 2008.